



Client Matter No. 80168.0118
Attorney Docket No. P5410

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Serial No. 09/754,155 Application of: Frank L. Weil et al. Filed: January 4, 2001 Art Unit: 2172 Examiner: Chongshan Chen Attorney Docket No. P5410 For: Search Engine Interface and Method of Controlling Client Searches	Confirmation No.: 3195 Customer No.: 32658
---	--

DECLARATION UNDER 37 C.F.R. § 1.131

MAIL STOP AMENDMENT
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

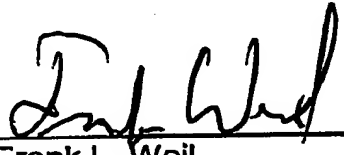
1. This Declaration and the accompanying documentary evidence are being submitted to establish conception and reduction to practice of the invention claimed in the above application in the United States at a date prior to August 31, 2000, the publication date of International Publication Number WO 00/51031 ("Hendren et al.").
2. I am a co-inventor of the invention recited in pending claims 1-20 and 22-24 of the above-referenced application.
3. Prior to August 31, 2000, my co-inventor and I had completed the invention recited in the pending claims 1-20 and 22-24. Design and implementation documentation for our search engine are included with this Declaration to show that the claimed invention was completed and in our possession. Additionally, the invention of claims 1-20 and 22-24 is shown to be

reduced to practice by the enclosed class diagrams, which illustrate one embodiment or design of our invention. The documentation and class diagrams were prepared prior to August 31, 2000. Additionally, the dates deleted from the enclosed documentation and class diagrams are prior to August 31, 2000.

4. At the time I conceived of and reduced to practice the invention recited in pending claims 1-20 and 22-24 of the present application, I was working for Sun Microsystems, Inc. at its facility in Broomfield, Colorado. I made the invention at my place of employment in Broomfield, Colorado.

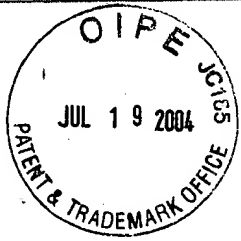
5. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon.

Date: 7/19/04


Frank L. Weil

Sun Educational Services
Product Engineering

WLS P2 / Search Engine
Document Last Modified: [REDACTED]



Project Initiator Search Engine

Frank Weil

RECEIVED

JUL 23 2004

Technology Center 2100

[Problem](#) | [Product Requirements](#) | [Expected Use](#) | [Solution Strategy](#) | [Dependent Groups](#) | [Constraints](#) | [Risks](#) | [Glossary](#)

1.0 Problem

An essential component in the evolution of WLS as a total online learning environment is the idea of WLS as a learning reference where a student can come back to over and over again. We don't want the student's learning experience to end when they complete a course. A key component of this reference model is a search capability into all of the resources and references available to the student. Searching encourages the student to visit WLS in hopes of finding quick answers to their questions.

2.0 Product Requirements

The following requirements are based on the [Web-based Learning Solution Phase 2.0 \(WLS\) - Marketing Requirements Document](#) and on discussion with Marketing.

Requirements marked with "high" priority are slated for first delivery, "low" priority for subsequent deliveries.

Req. #	Requirement Description	Priority (high/low)
1	Will provide a textual search.	high
2	Will search course content.	high
3	Will search course glossary.	high
4	Will search FAQs.	high
5	Will search discussion forums.	high
6	Will restrict searching to only students with an active WLS login.	high
7	Will provide a contextual search, where student can type a phrase or question and search will find "relevant" references (MRD 2.3.2, req #3).	low
8	Will provide option for student to choose services (ex. Course, FAQ, Discussion Page) to search (MRD 2.3.2, req #6).	low
9	Will provide option for student to search only currently active service (ie. search just the	low

	currently launched course).	
10	Will provide option for student to search all services to which student has subscribed.	low
11	Will provide option for student to search all services in WLS. Search hits outside of subscribed services would be displayed for informational purposes only. Student would not be able to visit those hits.	low
12	Each search request will search a course's content, glossary, FAQ and Discussion Forum together regardless of the origin of the search request.	high
13	Will display search results sorted by relevance.	high
14	Will provide viewing of glossary entries from search results without the need to launch a course.	high
15	Will support initiation of a search request from within course content, FAQ, and Discussion Forum.	high
16	Will support initiation of a search request from within the student administrative interface.	low
16	Will display with each search result, the type of service (content, FAQ, DF), course name of service, and relevance of search result (ex. percentage).	high
17	Will search WLS Online Help topics.	low

The following process flows are viewable forms of the process flows contained in the MRD:

Student Search Process

Subscriber Search Process

3.0 Expected Use

Students will be able to quickly perform searches within the service they are actively viewing, without having to necessarily go to a different page to perform the search. In addition, they will be given the option to perform more complex and expansive searches across other services. The results of the search will be displayed in such a way that the student knows, before selecting a link, which service the result came from (ex. A course, FAQ, Discussion Page, etc). Student will be able to select any of the search results if they are currently subscribed to the associated service. Selecting a link will take them to the appropriate location within that service. Student will be able to get back to the search results to select additional links.

4.0 Solution strategy

A solution strategy needs to address the following issues from the student's perspective:

1. Provide mechanisms for student's to submit search requests.
2. Efficiently search all services requested and adhering to any applicable restrictions (student authentication and authorization for services searched).
3. Display search results in a useful format and in a meaningful context. Control the number of "hits" displayed on any one page.
4. Correctly navigate to, and display pages in response to student selections from search results.

We are investigating third party Search Engine products to solve at least issue 2 above. Search Engine products may also help address issues 1 and 3 above, but will not address issue 4. Other strategic issues to consider for a Search Engine are

its ability to integrate with our current user authentication and authorization schemes, and our current page generation schemes.

The leading candidate at this time is Infoseek's UltraSeek. Sun has a site-wide license for the product and it is currently in use across Sun, including the JDC. There are downsides to UltraSeek that need to be evaluated and justified before formal selection can be made.

With the use of any third party Search Engine comes the need for the Engine to build index databases for anything that is to be searched. This will happen unknowingly to the student and presents administrative issues for WLS. Indexes need to be kept up-to-date either through nightly updates of the index databases or through some on-demand updating mechanism. These decisions may vary depending on the service.

One goal of the solution is to minimize the dependencies on third party products. The more we rely on the product for features the less flexibility we have in the future to expand or change our Search capabilities.

5.0 Dependent Groups

The following groups will be required to successfully deliver this project.

Group	Purpose
WLS Engineering	Due to the dependencies that the search engine will have on the entire WLS product, all members of the WLS Engineering team will need some level of involvement.
Product Support	With use of a Search Engine comes additional questions and issues from users on its usage. Troubleshooting and FAQ pages, and Discussion Forums may be needed specifically for the third party Search Engine product that gets deployed.
Publications	WLS user documentation will be updated to include a description of Search Engine functionality and usage. When help is made searchable, the help content may need to be augmented to facilitate searching (ex. HTML META tags).
Quality Assurance	All features delivered as a part of this project will be qualified. It is anticipated that the majority of this effort will be in testing points where WLS interfaces / integrates with the search engine.
TBT	Students will need to be able to submit search requests directly from the course interface. This will likely require a change to the course user interface. This will affect both JavaTutor and DES titles. It is not clear at this point whether there would be any need to change course content to better enable searching. Should "no mods" be a constraint?
Deployment Team / SunIR	If a third party solution is implemented, the deployment of WLS will also require the deployment of this additional software. It is unclear at this time what this means in terms of impacts to groups and processes. This will need to be investigated further.
Usability	This project will include new user interfaces. All new interfaces will benefit from usability testing. To optimize testing resources, it may be convenient to combine testing of Discussion, Search, and FAQ.
Database Administration	Given a specific course, Search will need to be able to identify its content, FAQ and Discussion Forum in order to restrict the search.

System Administration	If a third party solution is implemented, SA support will be required to investigate impacts to hardware and deployment configurations.
Legal	All changes to existing user interfaces and new user interfaces will require legal approval. Additionally, if a third party solution is implemented, Legal assistance may be required to deal with licensing issues. Even if we go with a product that currently has a site-wide license at Sun, what are the impacts of deploying WLS outside of Sun?
Performance Engineering	Depending on the third party solution, performance limitations and restrictions will need to be identified prior to deployment into a production environment.

6.0 Constraints

- Search must integrate with Discussion Page project.
- Search must integrate with FAQ project.
- Time may be a constraint if we need to deploy before the Y2K freeze. If this is the case then we can define a minimal set of search capabilities required for this release.

7.0 Risks

- Dependency on a third party search engine product and the issues it creates with respect to deployment and administration. Additionally, we need to identify performance related restrictions and limitations that affect the administration and maintenance of the production environment. For example, UltraSeek may require restarting at some regular interval.
- third party Search Engine site licensing issues (if any) could impact deployment schedule.
- In order to display search results to students in a meaningful context, we may want to consider modifying TBT course content to provide the information needed by search engines. For example, one default "view" of search results is to display the HTML file's <TITLE> tag value as the title of that search hit. If the file is missing a <TITLE> then the file's URL is displayed instead. The latter would not be acceptable in WLS.
If we establish "no mods to content" as a constraint then the complexity of the project may go up.

8.0 Glossary

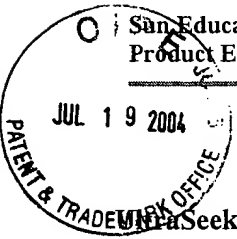
Course

A collection of services with a common learning objective and scope that a student can register for either in part or as a whole. Current course services include:

- Course Content (glossary is currently considered part of content).
- FAQ
- Discussion Page

Course Content

The authored course material organized as a set of structured HTML pages (ex. JavaTutor courses).



Search Engine Implementation Issues

UltraSeek General

1. How many Collections should we have?
 - One for all services
 - One per service type (Content, FAQ, Discussion Forum, Glossary, etc)?Collections are independently indexed. Collections offer a simple way to restrict searching. There is a limit (30-40) to the number of collections that can be searched at one time.
2. Can two installations of UltraSeek share a common set of collections?
3. Can the format of raw search result data returned be changed?
4. There are 3 log files: access, error, query. Are UltraSeek's logs automatically purged? Configuration issues?

Searching

1. On search results pages, can we include links similar to default UltraSeek results pages? Links include:
 - *search these results* to perform another search only on the pages in initial search results.
 - *Find Similar*

Indexing

1. Indexing Goals.
 - Content is only indexed on demand. When course is install, a request is sent to Ultraseek (Add URL option?) including the URL of the course. Ultraseek indexes the course at that moment. Additionally, we should be able to force a "revisit" of ALL courses. This may be necessary for maintenance reasons.
 - FAQ and Discussion Forums are indexed nightly during a specified time period. We could consider that indexing of FAQ and DF happen continuously 24 hours a day if we can ensure that normal student performance is not impacted.
 - FAQ and Discussion Forums URLs may not need to be "refreshed". We can handle refreshing manually if we can force a "revisit" on all FAQ/Forum URLs when necessary.
2. If an existing course is reinstalled with some existing nodes of the structure removed, how do we remove these nodes from the search index database? Can we clear out all links associated with one course?
3. Indexing Performance Need to determine the performance needs and options of the Ultraseek Spider. For each of the criteria below, capture %CPU usage, and Memory used (ex. use "ps -ofname,pid,ppid,vsz,pmem,pcpu,nlwp,s").
 - Difference between one indexing thread and many.
 - Difference between initial indexing time of files and updates, particularly for FAQ and DF.

WebX

1. Need JDC solution to indexing WebX via UltraSeek.
2. Will FAQ and DF share same top-level folder? One URL to index both?
3. Need some way to uniquely identify a FAQ or Forum during indexing. Need this identifier to be stored in Database. Options:
 - WebX forum id (assigned to the folder?)
 - Forum/FAQ title
 - Learning Event of course (All FAQ/Forums will not be at course level)
 - Internally generated ID that is included in Forum pages (meta tag).

Database

1. When indexing, courses can be identified by learning event id/version?
When a student wishes to visit a link we need to be able to map learning event id/version of a course to a student's registration id. Is this available in a current query (reg id => LE ID or vice versa)?

Search Software Architecture

Indexing Servlets

1. New servlets will be used for indexing

This indexing servlet will offer a limited API to support only the requests necessary to index Courses, Glossaries, FAQ, and Forums. HTML Pages returned by this interface will not be completely generated in the same fashion as they are through the normal student interface, but only as much as needed for indexing.

These indexing servlet will allow for specialized or different security and navigation rules.

2. Servlet can only index one course and one glossary can be indexed at a time.

In order to index more than one concurrently would require a form of session management that Ultraseek does not directly support.

Regardless, Ultraseek is only able to perform a single index request, at one time, to a particular site (ex. www.sun.com, java.sun.com, edutone.central.sun.com are sites).

3. Index servlet and session management. Index servlets cannot rely on HTTP sessions as a means to maintain service state (ex. Training Model for a course) throughout the indexing of links within a service. There is no guarantee that the indexing of a service from start to finish will occur without interruption. If the service is loaded and then Ultraseek or JWS is restarted then the session is lost and indexing will fail for the remaining unindexed pages. Since only one service can be indexed at one time state will be managed as follows:

Manage state statically within Index servlet. This is only necessary for indexing courses and glossaries.

- Each service-related index request will contain the service id of the service to launch.
- Index servlets will maintain statically the state of one launched service. When a request comes in, it is first authenticated (see Security, then the def URL or service id passed with URL is compared to the launched course state. If they do not match or there is no state then the course is launched and saved in servlet. If they do match then the request is processed.
- State will be maintained by a class implements the HttpSession interface so that this class can be used by our current page generator (which expects an HttpSession argument).

4. Indexing servlets will connect to db via the anonymous connection.

Indexing servlet will use this connection to retrieve all installed services.

5. URLs used for indexing differ from URLs students see in search results Since students access links using different servlets than indexing, the URL used to index each page will have to be modified before presenting to the user in a search results page. This rewriting of the URLs will occur in the index servlet before returning the requested page back to Ultraseek. Ultraseek allows the URL used for indexing to be overridden by a `url META` tag embedded in the page returned to the spider. The index servlet will insert this META tag with the URL needed by students to access the indexed page.

Course Indexing

1. Indexing servlet only needs to load course structure (via ModelReader) and not entire Training Model

There is no navigation or progress tracking needs, only a need to lookup URLs within the structure. However to help maintain a proper wrapper layer between ModelReader and end-user, a new IndexTrainingModel will be created. This allows other existing classes that work with TrainingModels to be used (ex. DP Page Handlers for page generation).

2. Course location string will be used as search result title

Ultraseek uses the `<TITLE>` HTML tag as the title of each search result. The Index servlet will insert this title tag in each page before the page is returned. The Course Locator string will be used as the title. This can be retrieved from the training model.

Forum/FAQ Indexing

1. See [WebX Indexing \(Forum/Faq\)](#).

2. Ultraseek will need to be able to talk to the WebX server, located outside of the firewall, the same way that the JWS needs to. If there is a proxy to route requests through then this is a configuration setting for Ultraseek.

Security

The following steps will be taken to provide for a secure indexing environment:

1. Install Ultraseek behind firewall and do not open up Ultraseek port to outside requests.
2. Ultraseek Admin account name and password will be defined as properties in lib/ts.properties with base64 encoding (same as is used for HTTP authentication).
3. All indexing requests initiated by Ultraseek will include a "UserAgent" HTTP header field with a value matching the Ultraseek Admin account password defined in lib/ts.properties. It is the responsibility of the individual index servlets and WebX templates to only accept index requests that contain this header.

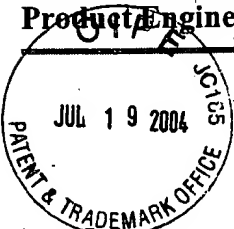
Initiating Indexing

- Courses, Glossaries and FAQs will be indexed by two different mechanisms:
 - a. Automatically as part of a nightly scheduled index request that is sent by the JWS to the Search Engine.
 - b. Manually through an administrator's request available from the Admin I/F.
- Discussion Forums will be indexed by two different mechanisms:
 - a. Automatically as part of a pre-defined indexing interval defined in the search engine. Default will be every 2 hours.
 - b. Manually through an administrator's request available from the Admin I/F.

Ultraseek Search Engine

- Four Collections: Course, Glossary, FAQ, Forum
- Searching just FAQ, Forum, Course, or Glossary can be done by including search criteria restricting results by collection name.
- Searching a specific service can be done by including search criteria restricting results to URLs containing the specific service id.

Best Available Copy



Search Deployment Architecture

Architecture

1. Use Third Party Search Engine, Ultraseek for performing searches on Content, FAQ and Discussion Forums. See evaluation.
2. Search Engine can run on same or different machine as WLS.

The decision of machines cannot be easily answered at this time looking just at Ultraseek. The attempt here is to provide information that can aide in that decision.

Either way, there are issues with installing and configuring Ultraseek. Configuration of Ultraseek is very similar regardless of which machine it runs on. Running Ultraseek on a separate machine may add additional issues regarding machine-machine access (http vs. https?, authentication?), but running Ultraseek on the same machine adds risk of ensuring that it doesn't take too many system resources away from WLS processes.

See Minimum System Requirements and Performance for information collected to date. We are in the process of collecting statistics more specific to the indexing of course content in an effort to understand CPU and Memory usage, but even these early findings will be on an Ultra 10 and not the production machine.

3. Ultraseek indexing requests are HTTP.

For Course Content, indexing requests will be routed through JWS. Ultraseek will not have direct HTTP access to Course Content files.

- o WLS can control what gets indexed.
- o Control of indexing authentication/authorization.

For FAQs and Discussion Forums, indexing requests will go direct to WebX after an initial request to the JWS to get the master list of FAQ or Forum links.

4. Student search requests are HTTP and will be routed to JWS. JWS will forward requests to Ultraseek.
 - o Allows for WLS authentication of requests.
 - o Enables restriction of search to only registered services.
 - o Hides Search Engine implementation from client. Allows WLS to search different services by potentially different methods (multiple search engines, DB queries, etc).

Minimum System Requirements

The following represents the minimum requirements of Ultraseek and should give a "ballpark" indication of our needs. We need to estimate what our anticipated Course Content, FAQ and Discussion Forum usage will be and perform some more extensive tests to define proper settings.

Computer	Sun SPARC Family Workstation or Server, or Intel compatible PC with Pentium class processor
----------	---

Operating System	Sun <ul style="list-style-type: none"> • Sun Solaris 2.5.1 and above, • Solaris 2.6 • Solaris 7
Physical Memory	<1000 documents - 64 MB <10,000 documents - 128 MB >10,000 documents - 256-512 MB Requirements may vary depending on number of collections and query rates
Available Swap Space	<1000 documents - 128 MB <10,000 documents - 256 MB >10,000 documents - 512 MB - 1 GB Requirements may vary depending on number of collections and query rates
Disk Space	10 MB for Application Index space requirements vary based on number of documents, size of documents, and content type. Generally, Ultraseek Server requires 10KB per document of free disk space
Network	TCP/IP over local or wide area network

Performance

What CPU and Memory usage issues does Ultraseek introduce?

Indexing (Search Spider)

1. Course and glossary content only needs to be indexed once upon its installation.
2. FAQs only need to be indexed whenever FAQ content is updated. This more frequent than Course content but not as regular a change as Discussion Forums.
3. Discussion Forums are updated on a regular basis so it should be indexed frequently (at most daily).
4. Rate of indexing is tunable (delay time between pages). This is a factor to consider if indexing occurs all the time versus during off-peak only times.

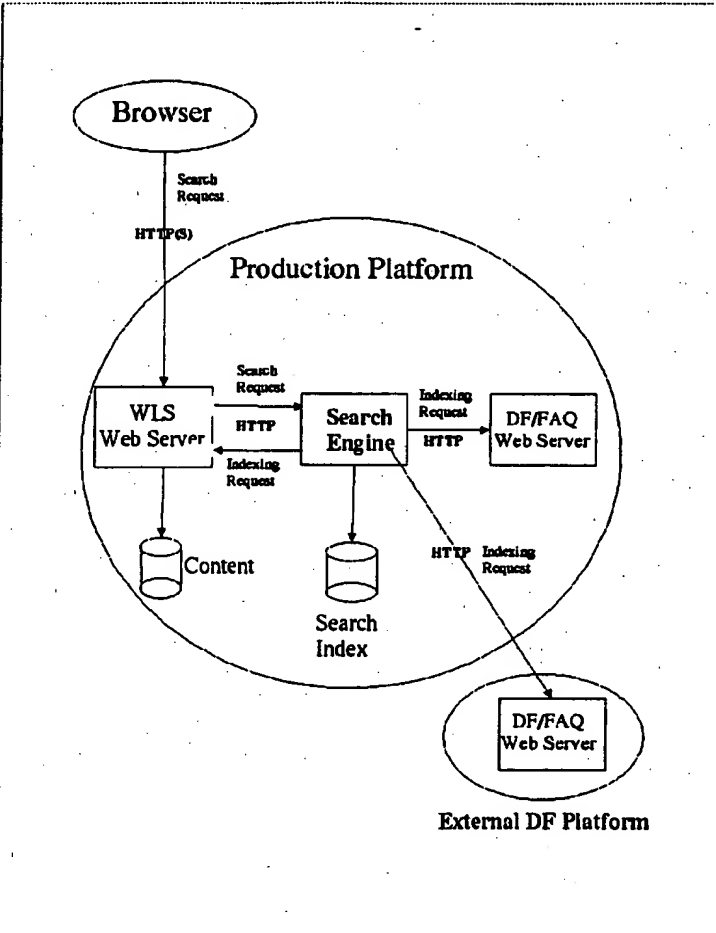
Student Search Requests

1. A student is either issuing a search request or a course navigation request, but not both at the same time. Assuming a "comparable" load for both (has not been measured yet) then searching does not introduce any significant degree of increase in load.
2. Number of search threads is tunable (default is 5). Impact is how long it takes for a student to get a search result response.

Architecture Diagram



Architecture Diagram





Search UI Prototypes

Search Request/FAQ/DF From Course

The following prototypes show options for displaying a search input area from a launched course. In addition to search, a link to FAQ and Discussion Forum for course are also prototyped.

Search options can be removed if search is not delivered at the same time as FAQ and Forum (same goes for FAQ).

Option 1 (Frame Version)

The FAQ and Forum link and Search area are in a new frame below the course frame. If too many more services are added later than can fit in the frame area, we can add a drop down menu for service selection.

Pros

- This overlay results in NO impacts to course templates or content.
- Allows us to format the look of these WLS options independent of the course look and feel.

Cons

- Takes up additional vertical real estate. this addition can be lessened if the course copyright is taken out of a frame and placed back in the course page.

Option 2 (Navigation Bar Variation 1)

FAQ and Forum Link and Search Area are in the same vertical toolbar as the course buttons. There is only minimal impact to course developers as these WLS features are "imported" into the courses templates from a Engineering-owned file. If too many more services are added later than can fit in the toolbar then either scrolling will occur in the toolbar area, or we can add a drop down menu for service selection.

Notes: FAQ and Discussion Forum help buttons are temporary and require more polished replacements or alternatives.

Pros

- Doesn't increase real estate needs.

Cons

- Links for course and for WLS features too tightly coupled.
- One time impacts to course templates.
- Engineering-owned template may need to stay in-sync with any look and feel changes made to Course templates. This could get complicated if ALL courses don't share a common look.

Option 3 (Navigation Bar Variation 2)

Same as Option 2 above with the following addition:

In order to give the impression that these new options are not course-specific buttons, they are separated from the course buttons.

Notes: FAQ and Discussion Forum help buttons are temporary and require more polished replacements or alternatives.

Pros

- Doesn't increase real estate needs.
- Clear delineation between course links and WLS links.

Cons

- Disjointed look to navigation bar.
- One time impacts to course templates.
- Engineering-owned template may need to stay in-sync with any look and feel changes made to Course templates. This could get complicated if ALL courses don't share a common look.

Option 4 (Search Request field on separate page)

Instead of allowing user to enter search request directly on course pages, the Navigation bar would have a search button that, when pressed, would display the search input screen in a separate window (SCAT window or a search-specific window). User then goes to this window to enter search requests. Search results are displayed in same window as search request.

This approach is similar to how Discussion/FAQ are implementing search input today.

Pros

- Simplifies changes to, and look of course window.
- Same window can be used for entering search requests for courses, FAQs, and discussion forums.

Cons

- Extra level of indirection in order to initiate a search.

Option 5 (Navigation Bar Variation 3)

Recommendations

Option 4 was the recommended option.

Search Results Page

Available Soon

Search Use Cases

Use Case	Retrieve Search Results
Description	Student displays a list of references to topics or phrases within a Course's content, glossary, FAQ and Discussion Forum as a result of a search.
Preconditions	Student is logged into WLC
Primary Scenario	<ol style="list-style-type: none"> 1. Student types a text string into the <u>search page</u> and hits carriage return. 2. System verifies student is logged in. 3. System searches course's services. 4. System displays <u>search results page</u>
Alternate Scenarios	<ol style="list-style-type: none"> 1. Student is not logged in. System displays "invalid session" page. 2. No result found for search. System displays Search Results Page with a "no results found" message in place of the list of search hits. 3. Student's subscription to all course services expired. System displays Search Results Page with a "no results found" message in place of the list of search hits. 4. Search Engine is not running (down for troubleshooting, upgrades, etc.) System displays "Search Facility Is Temporarily Down" page.
Variations	<p>Search Initiation</p> <ol style="list-style-type: none"> 1. Initiated from SCAT 2. Initiated from a Course Content page. 3. Initiated from a Course FAQ page. 4. Initiated from a Course Discussion Forum pages. 5. Initiated from the <u>search results page</u> <p>Services Searched</p> <ol style="list-style-type: none"> 1. System searches all services for a course (Content, Glossary, FAQ, and DF). 2. System searches only the services from which search was initiated.

Use Case	Visit Results
Description	System displays the page whose selected link was displayed in the search results.
Actor	System
Preconditions	After displaying the <u>search results page</u> the <u>student selects one of the links</u>
Primary Scenario	<ol style="list-style-type: none"> 1. System verifies student is logged in. 2. System verifies student is currently subscribed to course. 3. System displays requested page to student in window appropriate for service.
Alternate Scenarios	<ol style="list-style-type: none"> 1. Student is not logged in. System displays "invalid session" page. 2. Student's subscription to all course services expired. System displays Search Results Page with a "no results found" message in place of the list of search hits. 3. Link does not exist ??? 4. Student hasn't subscribed to service that page comes from ??? 5. Search Engine is not running (down for troubleshooting, upgrades, etc.) System displays "Search Facility Is Temporarily Down" page.
Variations	

Use Case	Search Help
Description	Student needs search help
Actor	Student
Preconditions	Student is logged into WLC and is viewing the <u>search page</u> or the <u>search results page</u>
Primary Scenario	<ol style="list-style-type: none"> 1. Student selects Search Help 2. Search Help is displayed
Alternate Scenarios	
Variations	

[| Introduction](#) | [Index](#) | [Features & Analysis](#) |

Section 1. Introduction

This document associates use cases with features that will be required to support these use cases. Also, first guesses of how these features might be implemented are suggested.

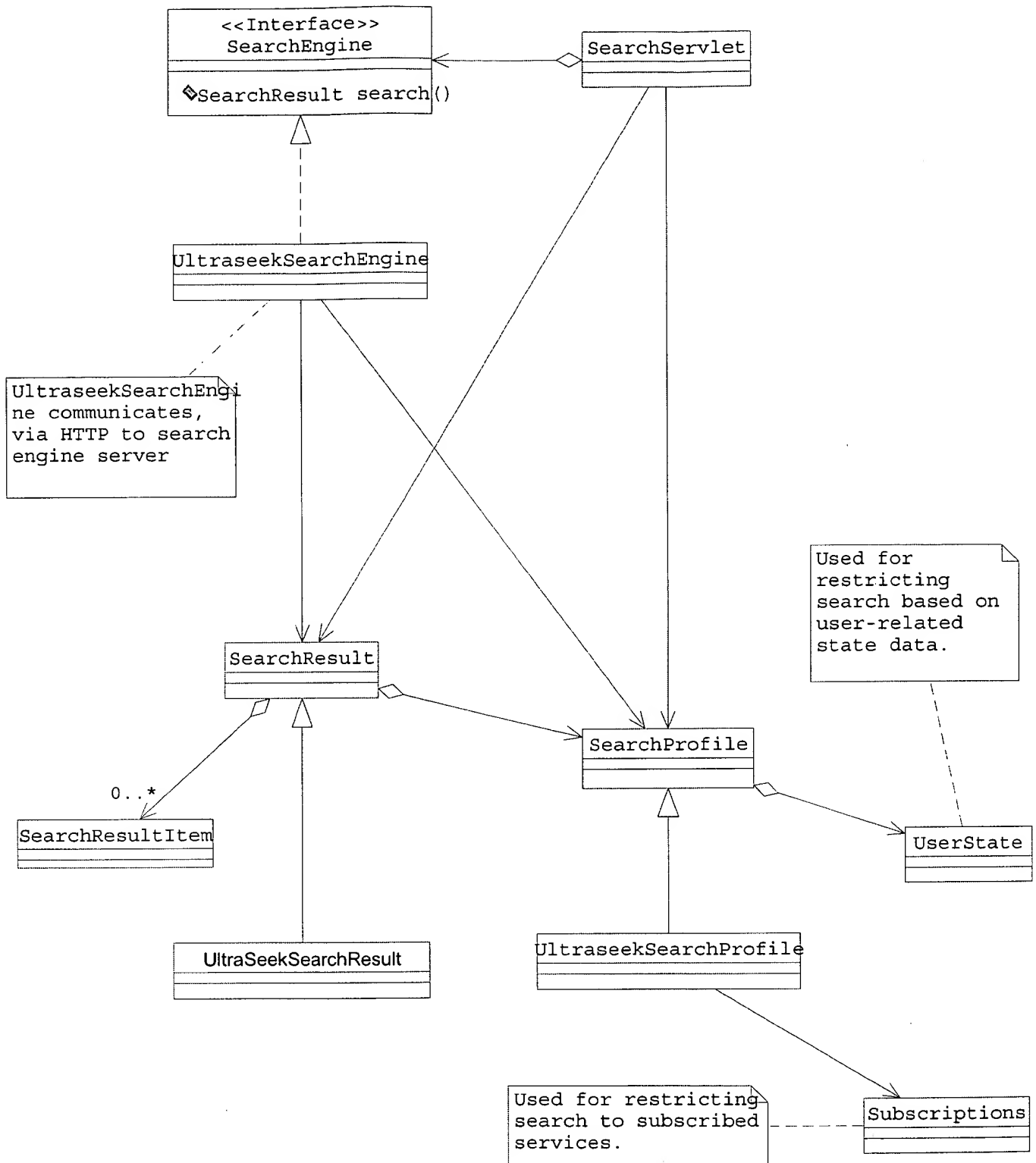
Section 2. Index

Features

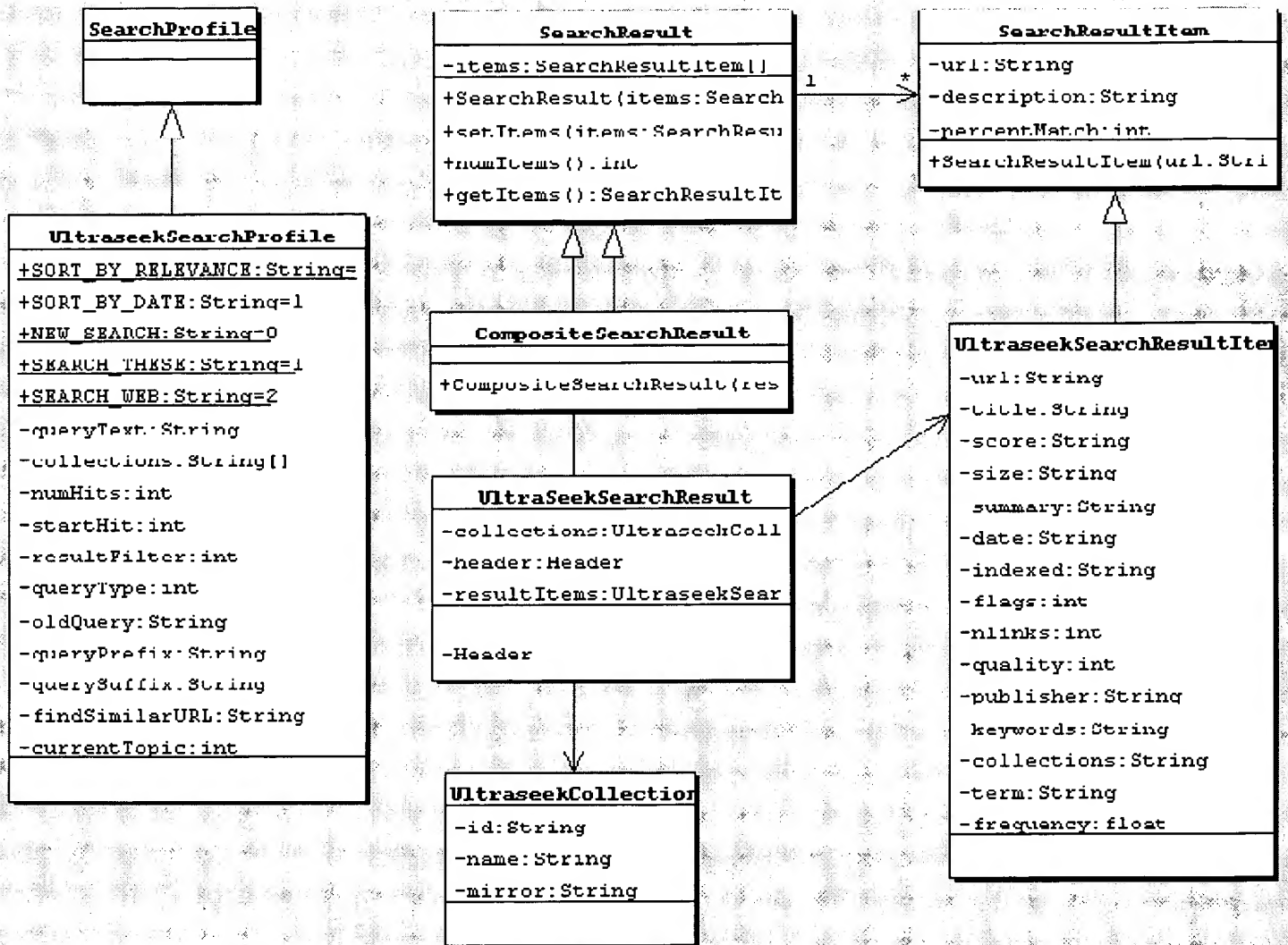
- Control the scope of searches
- Show results in groups of ten or so
- Search discussion pages
- Launch a service
- Verify the user is registered for a requested service
- Authenticate user's WLS registration
- Control our UI presentation

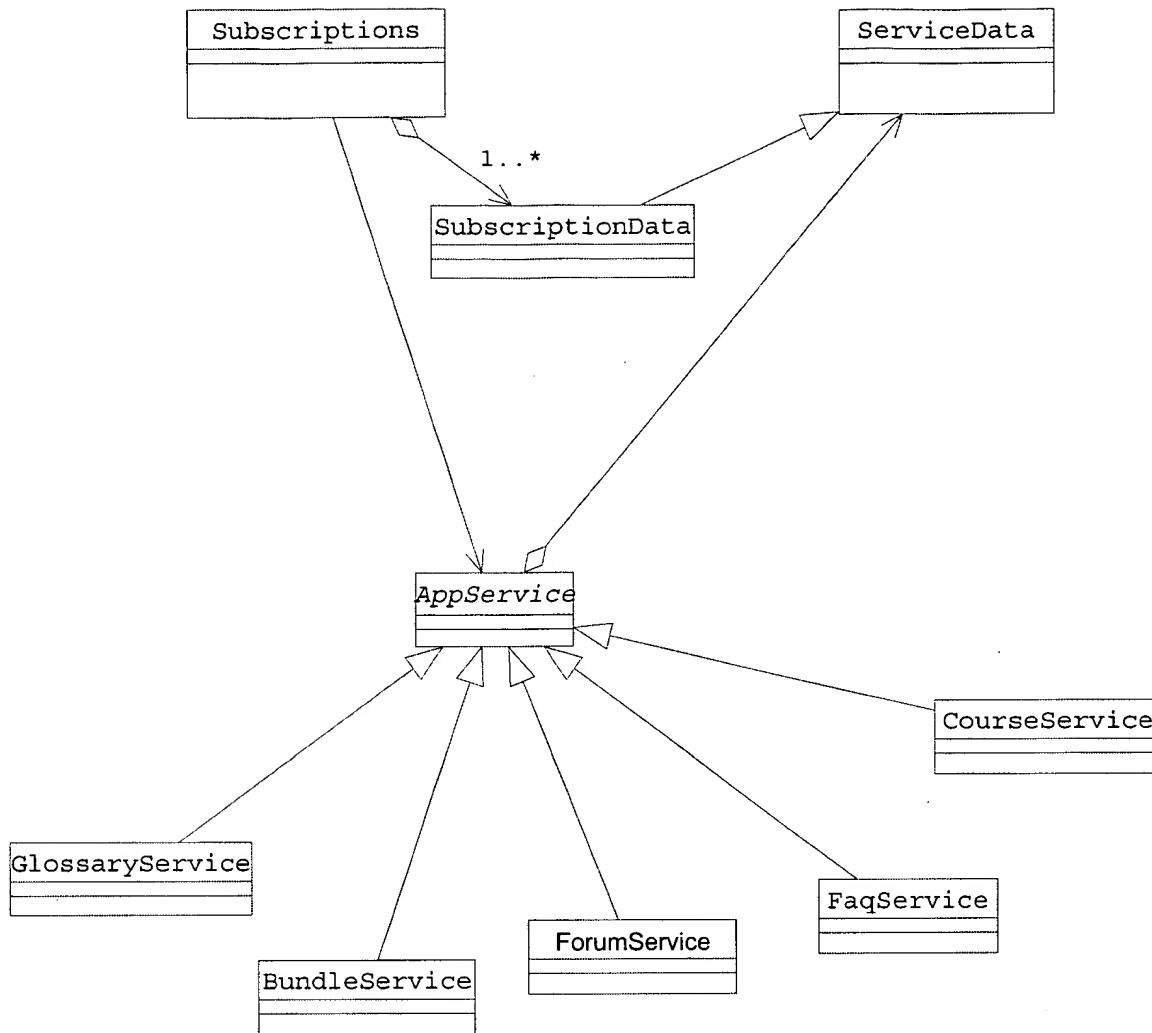
Section 3. Features & Analysis

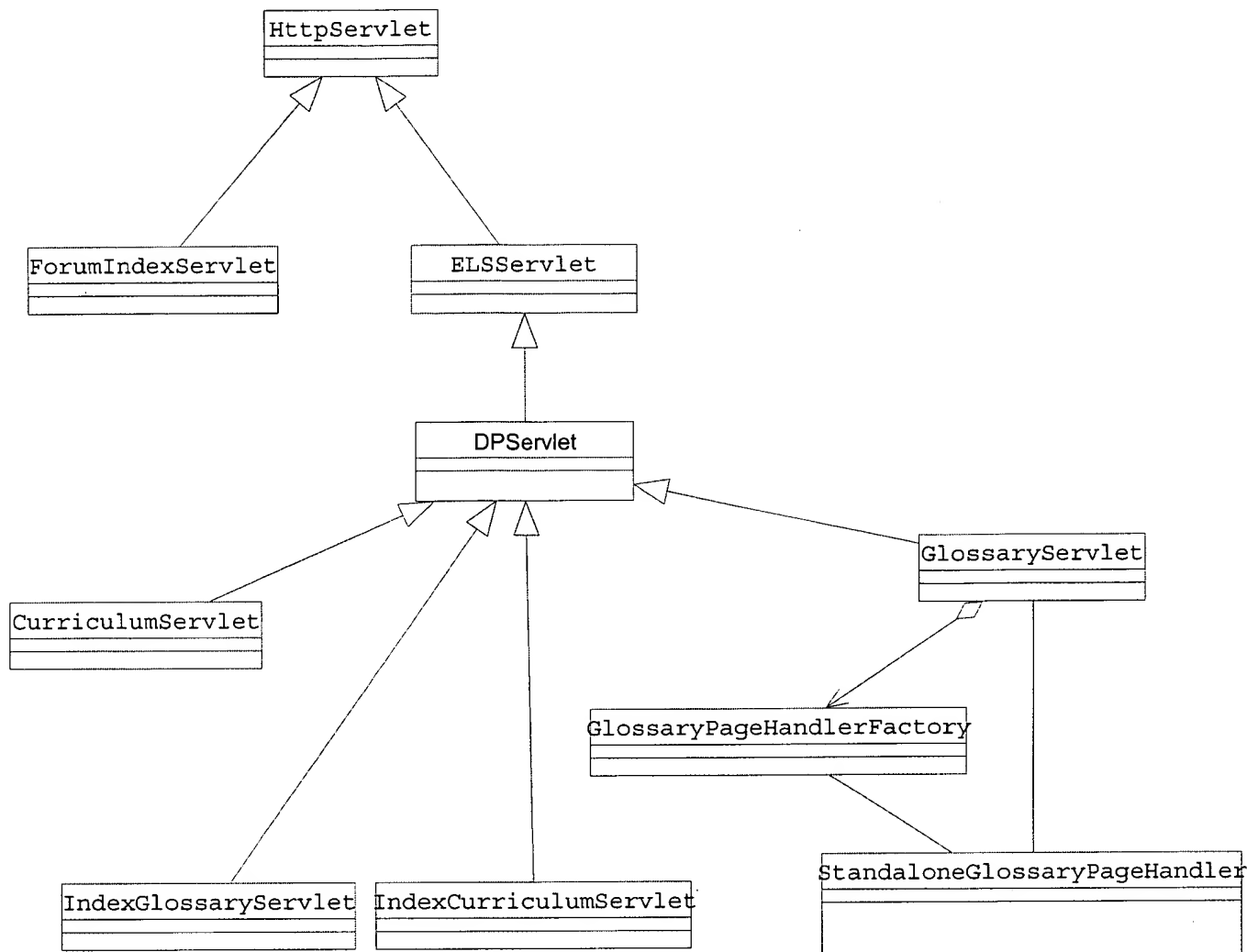
Feature Analysis	Use Cases 1-5, 11-16
Features	1. The ability to scope a search to the current services, all instances of the current services registered for, all instances of the current service, all services registered for, and all services.
Implementation	1. If UltraSEEK is used, it has the ability to manage separate data stores and search them both independently or together in groups. If we give each instance of a service its own datastore, then we can search them independently or in arbitrary groups.
Feature Analysis	Use Cases 17
Features	1. If a large number of results are returned from a query, we may want to show them in groups of ten or so instead of all in one giant page. This will require the ability either to cache the results of a query and then show subsets of it or the ability to re-execute a query and return a specific subset of the results.
Implementation	1. UltraSEEK has two query parameters specifically to handle this: nh-number of hits show; st-starting hit number. Re-executing a query with appropriate values for these parameters does what we want.
Feature Analysis	Use Cases 3
Features	1. The ability to search the discussion pages. If the COTS product Web-X is used, we will need to be able to search Web-X pages.
Implementation	1. This could be a problem for UltraSEEK. I ran a test where I set the UltraSEEK spider loose on Product Support's demo Web-X server. The spider never terminated and ran my machine out of memory after about 24 hours. However, the index file that the spider built worked just fine. Further investigation is required.
Feature Analysis	Use Cases 7, 8
Features	1. The ability to launch a service from a search result.
Implementation	1.
Feature Analysis	Use Cases 9 and others
Features	1. The ability to verify registration.
Implementation	1.
Feature Analysis	Use Cases All
Features	1. The ability to authenticate users and/or check for a valid session. There should be no back doors through which one could access search with out being logged in.
Implementation	1.
Feature Analysis	Use Cases All
Features	1. The ability to integrate search in the WLS user interface and customize it appropriately.
Implementation	1. UltraSEEK's UI templates are customizable. However, the most flexible solution is to build our own UI and access UltraSEEK programmatically as a service. UltraSEEK has a feature to return an easily parsable data file instead of its normal HTML UI forms. We can parse this data file into objects, then use our own page generation system to present the data. This give us complete control over our UI.
Feature Analysis	Use Cases
Features	1.
Implementation	1.



Best Available Copy







Index servlets derive from DPServlet to share the common page generation methods. Eventually these methods will be relocated to non-servlet class then index servlets can derive from HttpServlet.